小白成长日记——第四天

我变强了，也要秃了

js 语法

- 在比较的时候 要用 ===
  NaN这个特殊的Number值与其他所有的Number值都不相等，包括自己 Nan === NaN //false;
  唯一能判断的是使用isNaN(NaN);//true;
  浮点数比较是也要注意：
  1 / 3 === (1 - 2 / 3); // false
  这不是JavaScript的设计缺陷。浮点数在运算过程中会产生误差，因为计算机无法精确表示无限循环小数。要比较两个浮点数是否相等，只能计算它们之差的绝对值，看是否小于某个阈值：
  Math.abs(1 / 3 - (1 - 2 / 3)) < 0.0000001; // true
- 数组创建：

```
var arr = [1, 2, 3.14, 'Hello', null, true];
arr[0]; // 返回索引为0的元素，即1
arr[5]; // 返回索引为5的元素，即true
arr[6]; // 索引超出了范围，返回undefined
```

- 对象：
  JavaScript的对象是一组由键-值组成的无序集合

```
var person = {
name: 'Bob',
age: 20,
tags: ['js', 'web', 'mobile'],
city: 'Beijing',
hasCar: true,
zipcode: null
};
```

  调用方法就用对象名.属性名：person.name; //Bob;
- 编写js要用static模式，用var声明局部变量；防止错误；
- ASCII字符可以以\x##形式的十六进制表示:'\x41'; // 完全等同于 'A';
- 多行字符串: 由于多行字符串用\n写起来比较费事，所以最新的ES6标准新增了一种多行字符串的表示方法，用反引号 ... 反引号 表示
- 模板字符串:如果有很多变量需要连接，用+号就比较麻烦。ES6新增了一种模板字符串，表示方法和上面的多行字符串一样，但是它会自动替换字符串中的变量;
  toUpperCase()把一个字符串全部变为大写：
  toLowerCase()把一个字符串全部变为小写：

indexOf()会搜索指定字符串出现的位置：
substring()返回指定索引区间的子串：

- 数组:
  与String类似，Array也可以通过indexOf()来搜索一个指定的元素的位置

  slice()就是对应String的substring()版本，它截取Array的部分元素，然后返回一个新的Array： 注意到slice()的起止参数包括开始索引，不包括结束索引。
  如果不给slice()传递任何参数，它就会从头到尾截取所有元素。利用这一点，我们可以很容易地复制一个Array

  push()向Array的末尾添加若干元素，pop()则把Array的最后一个元素删除掉
  如果要往Array的头部添加若干元素，使用unshift()方法，shift()方法则把Array的第一个元素删掉
  sort()可以对当前Array进行排序，它会直接修改当前Array的元素位置，直接调用时，按照默认顺序排序
  reverse()把整个Array的元素给掉个个，也就是反转
  splice()方法是修改Array的“万能方法”，它可以从指定的索引开始删除若干元素，然后再从该位置添加若干元素

```
var arr = ['Microsoft', 'Apple', 'Yahoo', 'AOL', 'Excite', 'Oracle'];
arr.splice(2, 3, 'Google', 'Facebook'); // 返回删除的元素 ['Yahoo', 'AOL', 'Excite']
// 只删除,不添加:
arr.splice(2, 2); // ['Google', 'Facebook']
arr; // ['Microsoft', 'Apple', 'Oracle']
// 只添加,不删除:
arr.splice(2, 0, 'Google', 'Facebook'); // 返回[],因为没有删除任何元素
arr; // ['Microsoft', 'Apple', 'Google', 'Facebook', 'Oracle']
```

  concat()方法把当前的Array和另一个Array连接起来，并返回一个新的Array；
  请注意，concat()方法并没有修改当前Array，而是返回了一个新的Array

```
var arr = ['A', 'B', 'C'];
var added = arr.concat([1, 2, 3]);
added; // ['A', 'B', 'C', 1, 2, 3]
arr; // ['A', 'B', 'C']
```

- 对象：
  要判断一个属性是否是xiaoming自身拥有的，而不是继承得到的，可以用hasOwnProperty()方法
  xiaoming.hasOwnProperty('name'); // true
- 条件判断
  JavaScript把null、undefined、0、NaN和空字符串''视为false，其他值一概视为true；

- 循环
  for循环的一个变体是for ... in循环，它可以把一个对象的所有属性依次循环出来,要过滤掉对象继承的属性，用hasOwnProperty()来实现：

```
var o = {
 name: 'Jack',
 age: 20,
 city: 'Beijing'
};
for (var key in o) {
 if (o.hasOwnProperty(key)) {
    console.log(key); // 'name', 'age', 'city'
  }
 }
```

  由于Array也是对象，而它的每个元素的索引被视为对象的属性，因此，for ... in循环可以直接循环出Array的索引：

```
 var a = ['A', 'B', 'C'];
 for (var i in a) {
  console.log(i); // '0', '1', '2'
  console.log(a[i]); // 'A', 'B', 'C'
 }
```

  请注意，for ... in对Array的循环得到的是String而不是Number

- JavaScript还有一个免费赠送的关键字arguments，它只在函数内部起作用，并且永远指向当前函数的调用者传入的所有参数。
  arguments类似Array但它不是一个Array

```
function foo(x) {
   console.log('x = ' + x); // 10
    for (var i=0; i<arguments.length; i++) {
       console.log('arg ' + i + ' = ' + arguments[i]); // 10, 20, 30
       }
}
foo(10, 20, 30);
```

- rest 参数
- return 多行的时候要加大括号并在最后加分号